Usage of SOA and BPM changes the roles and the way of thinking in development

A.Selmeci, T. Orosz
Óbuda University – Alba Regia University Center
Budai str. 45, H-8000 Székesfehérvár, Hungary
Applied Informatics Doctoral School
E-mail selmeci.attila@arek.uni-obuda.hu; orosz.tamas@arek.uni-obuda.hu

Abstract—**The processes in the ERP (Enterprise Resource Planning) systems were automatized in the past several years. The evolution of the automation started with process flows, followed by role based task separation. In the same time the systems started to decouple the monolith solutions to smaller pieces to create more flexible components from them. This approach leads the ERP systems to build and offer services for collaborative purposes. The web enablement provided more power with the Web Service technology to drive the technology to follow the SOA (Service Oriented Architecture) design paradigm. The services orientation step over the system and sometime the company boundaries and require integrations on different level. One of these integration layers is the level of composites, where new application above the services can be built. Another layer is the process engine, where the services can be linked together into enhanced workflows defining backend and human activities. This layer is defined as Business Process Management (BPM) for business application. This paper intends to show the requirement of SOA implementation requirements from human point of view and describes the different thinking in design, development and end-user, when BPM and composites are in use.**

Keywords: **Workflow, Composite, SOA, business process management, SAP, ERP, BPM, System Identification**

I. Introduction

This paper goes through tangential on the history of business programs and processes. Touching the human activities and requirements. We tried to collect the information from the mentioned areas to generate a more accurate synopsis on the given requirements, possibilities and solutions on an abstract level. Our study targeted to show the today's requirements for an employee, decision maker and of course for a solution. The behavior changes are very important because the applications and technologies are changing much faster than the employees' personal attitude and thinking can work with them or even follow them. The paper collects not only the business application evolution, but the parallel technology evolutions as well to make requirements changes more understandable. The main stream of the study is the changing and spread of the automation, and the decoupling and proportioning of the monolith processes to single services, and the influence of these changes on the person at a high level.

II. Single Processes

Business processes are the main driving flows in a company's life. The early ERP (Enterprise Resource Planning) systems targeted two main pain points of companies: real-time information and integrated systems. The business programs at the different companies solved separated problems, processes by isolated solutions. During that time (thinking about '80-s) the different parts of companies wanted to have their own solution, or better to say programs for register and manage their data to make easier the daily work. Of course the early programs could not solve the problems and made bigger issues and required more human activities to fill the program with data, check and verify the consistency, semantic and relevance of data, and of course manage the infrastructure. From technical point of view the backup, restore and recovery processes were very important, but very pure at that time.

From person point of view we have to differentiate only developer and end-user (the user we executes the tasks). The developer tried to understand the process what the end-user explained and immediately code it in a simple way. Of course the end-user was far away from the informatics and could not translate the requirements to a simple developer. Another, sometimes better way came in the game, when the end-user had some interest on programming. This approach led to better semantic result, which was the elementary important goal. System identification was not part of the application development; only some well-defined input-output data were given and checked. Definitely the main result was in focus, not the decision and execution path to reach it. These applications tried already to handle not only one single task, but cover more topics as well. As we uncovered the early codes were written by singletons having sometimes direct knowledge in a subject, but not in each touched areas. The customers pushed the programmers to create application working for different areas simultaneously. The simplest applications handled register and catalog requirements, and in a better case many of the followings: managing material master; inventory management; basic sales functionalities (goods list, prices, handling rebates, simple invoicing, following payment by prompts). These applications did not follow the World's trends, just wanted to solve some problems and issues at the customer. Of course the 60s and 70s brought some widening in western countries, where the current programming languages (e.g. ALGOL, COBOL, FORTRAN) made it possible to build better solutions. The eastern countries (eastern block) had some (many years) lag because they could not get or buy the western results or products. That's why this section is dealing with 80s (for eastern block) and 60s-70s (for western block) as well. The applications were very close to the database level, because they brought up to the end-userlevel. The three main data manipulation language elements (insert, delete, modify) were almost in the hand of users. The system level operations, checks, like output management were also part of the application logic.

From technology point of view the data storage is an important part of the solutions. In the early applications the data were stored in single files and each application implemented its own data store mechanism. These mechanisms could handle only calculations and simple data selections. The application based on this purpose could not really handle multiple users, because the data was stored in a single machine on the file-system. On the other hand it lead to a problem that a specific task required many updates in different files (Fig. 1).

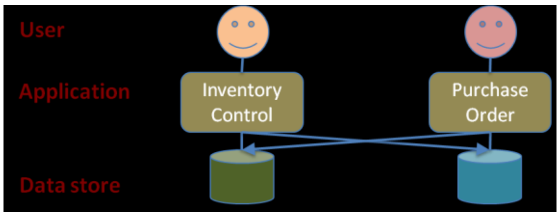

Figure 1 File access using file system storage

The file system storage raised the problem that many files are touched during a single business level task execution. This slowed down the whole procedure and generated other limitations as well: separated and isolated data lead to difficult representation and data coordination; data duplication in different files generates integrity problems with more space requirements; file changes require application level program modification; difficult file access control because no data sharing is available. These application were not so stable, reliable or even scalable. In such an application each task were started by a user and executed online. On the other hand these application were not designed as general solutions, but targeted to solve the single customer requirements.

III. INTEGRATED SYSTEMS, MONOLITH APPLICATIONS

Many economical results influenced the application designs. In 70a the production arise as a central point in such application. The main focus was not only to follow and register the movements, purchasing, production, sales, but manage the manufacturing. For that the material requirements planning (known as MRP 1, Fig. 2) became an object to solve. The application during that time tried to step over the simple inventory control (IC) package solutions and not only store and manage (insert, delete, update) the data, but calculate and plan the requirements from the known data.

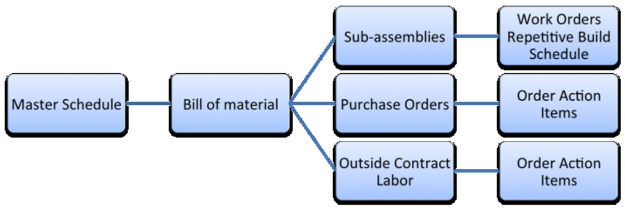

Figure 2 MRP 1 – Material Requirements Planning

To create such an application the original approach (having single programmers) was not sustainable anymore. Economical knowledge, abstraction capability, coding skills were needed and all these should have been coordinated. These lead to establish smaller or larger software companies focusing on business applications. In 70s the relational model came into play, where the data was separated into individual tables (entity types). Relation keys realize the relationship between tables. Initially required heavy system resources. Examples: Oracle, Sybase, Informix, IBM DB2 (Fig. 3).

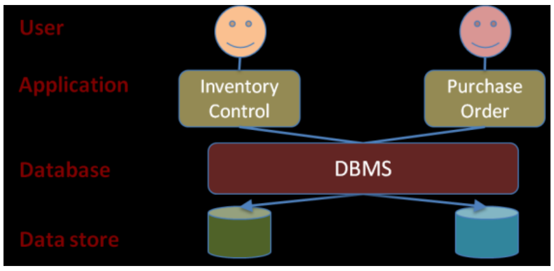

Figure 3 DMBS storage – Simple architecture

As we discovered earlier the early applications were one-time software and targeted one customer. Different people wrote different applications storing data in different way directly in files. Without knowing the file storage technique (idea implemented) no one else could enhance the application and the data could not be extracted. The database management systems intended to solve this problem as well, not only giving another layer. Other topics to solve are the following:

- In file storage technique redundant data storage and inconsistency could occur, the database management system keeps one copy of the data
- Data access depends on the file storage mechanism, but a query language for databases can standardize the access
- Constraints can help in integrity problems
- Files can be accessed by single users, but databases provide a multi user layer
- Data access rights can be added to improve security level
- Separated and managed data backup and restore mechanism.

The database management systems brought not only a middle storage layer, but an additional expertise requirement to implement an application using DBMS and it increased the initial cost as well. The business applications implementing the MRP requirements faced quickly with the lack of data integrity, because errors in the inventory or BOM (Bill of Material) data results error in the output. The basic processes were not controlled and flexible to follow the manufacturing requirements, like managing many factories, different locations, separated warehouses, having no calculation of amounts. All these (and other not mentioned) problems lead the business applications to the next stage where the planning was in focus. The 80s brought the manufacturing resource planning (known as MRP 2) optimizing manufacturing processes by synchronizing the materials with production requirements. The Figure 4 shows with different color the basic MRP 1 processes were extended with some steps to assist planners in tracking problems associated with inventory control. These steps are in most cases feedback information, which start a loop by rescheduling certain items. There are many systems providing the basic modules of MRP2 paradigm: Master Production Scheduling (MPS); Item Master Data (Technical Data); Bill of Materials (BOM) (Technical Data); Production Resources Data (Manufacturing Technical Data); Inventories & Orders (Inventory Control); Purchasing Management; Material Requirements Planning (MRP); Shop Floor Control (SFC); Capacity planning or Capacity Requirements Planning (CRP); Standard Costing (Cost Control); Cost Reporting / Management (Cost Control); Distribution Resource Planning (DRP).

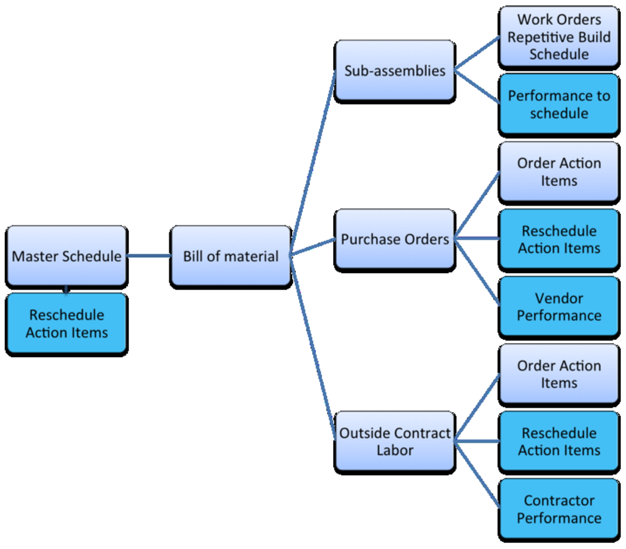

Figure 4 MRP 2 – Manufacturing Resource Planning

There are many other, here not listed modules in an MRP2 system, but the main financial modules should be accentuated: general ledger, Account payable and receivable, sales order management, etc. The MRP2 systems were the predecessors of the Enterprise Resource Planning (ERP) systems (Fig. 5).

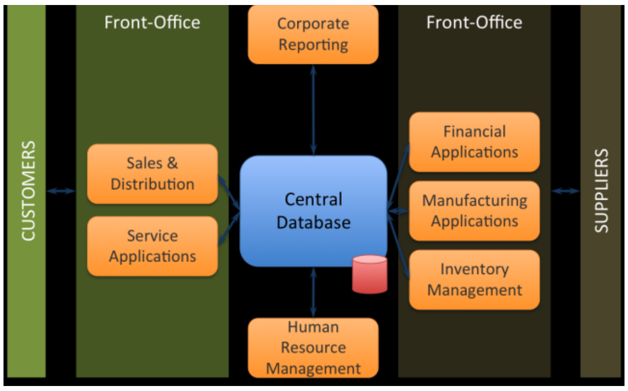

Figure 5 Enterprise Resource Planning system concept

The ERP systems first appeared at late 80s with power of enterprise-wide inter-functional coordination and integration. The last three major steps in business application were possible because the database management system made easier the separation from the file-system and integrate the data in a consistent and non redundant way. As the Figure 3 shows the multi user access and role separation was also available. The database management systems required much more computational power than the single purpose early applications, but offered many features for application programmers. During that time only main frames and mini computers were available for handling huge data amounts and calculations. SAP (Systems Application and Programming) one of the leading ERP software companies, was founded in 1972 by IBM engineers. The main idea from SAP was an ERP system, which is integrated and makes possible the real-time data access and processing. SAP's first software product name was R/1 followed by R/2 in six years. The letter 'R' in the name refers to real-time. The number behind shows the number of layers the system used. The main product of SAP in early 80s was the R/2 running on a mainframe machine having two layers: first layer is the presentation as a console, and the second layer runs the database and the application as well. The R/2 system was rather MRP2 system then a full ERP.

If we consider the roles during the product lifecycle as we did for single purpose applications (called single processes above), we can realize that there are many different roles here as the requirements grown. The developer is not a single person contacting the customer anymore. The different levels and modules of the integrated system require different expertise and knowledge. Such systems are designed from the main processes, modules point of view. These modules have definite subjects to implement according to the scientific results, but each of the customers wants to have some special requirements. For a so huge system built from integrated modules solution managers are needed to communicate with the targeted bigger costumers to establish a well-defined and approved solution plan. The solution must be translated for the developers; therefor a product owner should be involved. The solution manager can speak on the customers' language and the product owner understands the requirement having some real customer experience and architect skills to design the data and functional model. The developers receive their tasks on different level, like database design for implementing new data model keeping integration with other modules in mind. No duplication is allowed in the data model, but internal interfaces must be prepared and considered. It requires many communications on design and implementation level as well. Beyond the data model developer we should calculate with business logic developer and User Interface (UI) designer. In the early monolithic systems, like SAP R/2 the UI design and the business logic run together, there was no separation. We can say that SAP has a unique idea for software development, because SAP evolved an own internal programming language called ABAP (Advanced Business Application Programming). SAP defined two UIs: reports and dynpros (Dynamic Programs). Input enabled screens are always dynpros and the data input is given through SAP transaction having several dynpros. Each of the dynpros defines its business logic, called flow logic, which is strongly built together the UI layout. The reports theoretically read from the database, calculate something according to the business needs and presents the result as a list output. This software development idea where the application is not depends on the hardware, on the operating system or even on the database management software, brings stability, sustainability for the applications developed in the environment. It makes much easier to plan for longer time and design development object for future use as well. This can lead us to reusability observing some conditions.

As we have seen this kind of applications tried to put everything together and integrate them into one single database. This integration wave had an effect on the small single process applications as well, because the developer tried to put more processes, modules together into one database. The limitations of some technologies could not let the developer to grow over the capabilities and capacities, which pushed some solutions out of the market. On the other hand different approaches of diverse solutions could not be smoothly fit into one database as an integrated whole. Which means it was easier for those who tried to put pieces together to redesign each solution pieces again under one common umbrella.

From processes point of view this era evolved not only monolith holistic systems with several business transactions, but brought immediately the need of activity group separation. In a single process system usually one user executes everything (each of the given, low number of processes). In such a large, integrated, multi module system the tasks should be separated, to be able to follow the data flow. The business processes were not clearly defined, only the successive steps were defined according to the customer needs. One step could be a transaction, a report run, or even execution of a batch job, etc. The process flow was not written or driven, but only in brain controlled. It was only process based thinking or workflow like thinking in mind. If we try to follow the timeline, this kind of process control was around early 90s. The next decade brought a huge change in process control. Many of the ERP providers tried to build a workflow system to handle the process flow requirements coming from the market. We can speak about technical workflow systems, which only offered a process engine for general purpose. Many companies decided to use such solutions, but the integration with the existing ERP solution was always a problem, even if the workflow engine was a best of breath category. The other approach for creating automated process flows is the built in workflow system. Basically both directions required the same from the ERP system, namely giving opportunity to access the single steps (report, transaction, etc.) available. Although it was an extraordinary idea, task and brought great solutions, but from the other point of view it moved the process control status only a bit forward to the goal. This small step is that the single process steps remain the same in the system, but the workflow gave automation above them.

If we consider SAP as a good example for internal workflow solution, we have to know something about the Business Framework Architecture (BFA). SAP implemented this framework as thinking, and as a new layer above the standard process steps (transactions, reports, etc.). The BFA defined logical groups called business components following the module separation defined already in the integrated ERP system. These components grouped so-called business objects together. For easier understanding as an example we can consider financial services as a business component and invoice could be a business object in it. The business object tier is an object-oriented layer for internal and external usage as well. The methods of the business objects embed the standard process steps, like approve invoice, or park an invoice, if we follow our previous example. The business objects have of course attributes for keeping the status of the given object (e.g. invoice), some of them are special ones, called key attributes just referring to the underlying data model. The business objects can publish their status changes by means of events. If an invoice is approved an event is generated. If we put the pieces together we have to consider that each step of a workflow is a method of a given business object holding its state in its attributes. Of course the workflow engine, called in the SAP world Business Workflow, has many standard control functions, like cases, loops, etc. The most important thing is the event of the business objects, because a workflow can be started on an event or a workflow can stop at a certain point in flow and wait for another event, triggered even by another workflow. With this scenario SAP implemented a workflow engine, which is really business driven and not only a technology offered for others. If we consider an external workflow engine, it can use the public (externally callable) methods of SAP Business Objects, called BAPIs (Business Application Programming Interfaces) to execute process steps within the SAP system according to the defined process flow.

Summarizing the topic the integration brought up the role separation and the automation of the processes. The companies implementing an ERP solution have high level and detailed process knowledge. The single process application, detailed in previous section could solve only individual problems. At a company many different applications were implemented without human communication. Individual application was brought into action for single person (or department) requirements. Parallel activities were executed in different applications having diverse process flows and ideas behind. Only some clerk knew the usage of the single application. Even they knew their business processes well, they could not change positions easily without knowing the other kind of application. A company using ERP solution generally tries to rationalize the processes. It results two important changes within the company: clearer, more controlled and standardized processes are available so an ERP application can be implemented easier and not too many modification are required in the delivered software; smaller number of users is required to feed the system with data.

These changes cause cost saving for the companies. On the other hand the roles and a bit the behavior of the users (employees) are changing if workflow is used. The workflow steps are measured, expected and even maximum execution time can be assigned with alerts as well, the different departments should not wait for the others to get the information for the next steps, in case of vacations substitution can be used as well. These speeds up the whole process, the users can execute much more tasks during the day and the company generates more revenue. In the early systems (around early 90s) there were no separation on the roles and tasks, only single persons executed the required steps. In the later workflow solutions the roles are separated and different users execute the different tasks. This work distribution was a main step in the business process flows.

IV. Communicating Applications

Some of the ERP solutions were not really integrated into one database. There are solutions having modules (sold separately), which have many connecting installations. Following this approach we see that this makes easier:

- The handling of the purchased modules from sales point of view.
- To connect the system with others. The communication is daily work, so easier to interface the modules with external solutions.
- To make distributed environment with different locations (because of the interfacing).

On the other hand this kind of solution generates higher operating system, database management system, communications costs because of the number of component. The data is duplication in the whole environment many times.

One of the examples for this approach is the Nexon's HR solution. There are different modules in different installations for e.g. human management, payroll, time and event management, benefit handling, education.

These kinds of solutions frame a small but interesting group. If we consider the roles at a company using such a solution, we realize that the end-user do not know whether the information is coming from and integrated single database system or from a distributed system if he or she is working in a module only. If more tasks are relevant using different modules, common surface and single sign-on is required. The common surface, like a portal can obscure the system distributions separated at the module edges.

V. Single purpose solutions

Around in the same time the market wanted special solution for special purposes. It means the company needs step over the boundaries of ERP making the best of early Internet. The new applications pointed again smaller, but specialized processes like the early single process applications. The difference was that no basic functions were required in the application anymore, because those were implemented in an ERP system already, but the application could focus on the special topics. If we consider the separation, immediately the communication comes into focus. The ERP system and the early MRP solutions had different communication possibilities, but the Internet capabilities opened a new dimension to communicate not only between systems in the data center, but via the Internet with partners. This was the beginning of e-business. The basic, standard processes stabilized by the ERP solution, which made possible for earlier not so important functions based on the standardized ERP processes to came into focus. Such areas are the supplier services, handling customer data, or even executive reporting. As an example an ERP system should handle the customer master, but in the delivery process the customer is one of the main important information, but an ERP is not dealing with the marketing, the sales channel opportunities, even call centers, problem solving and other front-office processes. This leads the market to build new solutions for the different newfound tasks. The roles were changing here again, because the earlier not so significant tasks became important and new departments and systems for them were required. The people working in these areas do not need to know the main ERP processes; they can focus e.g. on purchasing, the customer only and record the information in the system, etc.

As we have an example software vendor used in this paper we can continue with that again. SAP realized at the end of 90s the market needs and lunched in 2000 the new dimensional products. These products were not so stable not so perfect as the ERP solution, but the customer requirements were also not so clear. After some years SAP as other software companies learned what kind of areas could be interesting for the customers and with these knowledge rebuilt the applications and created real new solutions, like CRM (Customer Relationship Management), SRM (Supplier Relationship Management), PLM (Product Lifecycle Management), BI (Business Intelligent). Each of these solutions covers not only a single application or system, but a solution with one or more component depending on the processes used.

From our perspective these solutions have to communicate, share or duplication data. The huge communication requirements pushed the vendors to use standard interfaces and make it possible really to use the Internet as a communication layer. The communication between systems can be different from technical point of view, but for business application the business content is important, so application level, loosely coupled linkage were needed. In the subject of EDI (Electronic Data Interchange) the EDIFACT was the standard developed by the United Nations. The main purpose was to give a message based structured format for electronic communications for commerce and transport. The EDI as idea was implemented in many solutions, but definitely not the EDIFACT (Electronic Data Interchange For Administration, Commerce and Transport) standard. For example SAP implemented its own ALE (Application Link Enabling) solution, which is a message (called IDOC, Intermediate Document) based business content communication. SAP uses this messaging for distributed solutions as well. Many vendors (but not SAP itself) built converter, so-called EDI-interface for SAP to translate the SAP IDOC format to EDIFACT. The other important direction from the paper point of view is the process management. Each of the solutions has their own workflow solution for the processes inside. But there are some processes, e.g. sourcing, where processes go over the edge of the solution. Practically the employee selects from the catalog a good (or material) and wants his manager to approve it. This workflow is executed within the SRM solution. But the approved purchase requisition must go into the ERP system for other steps to create a real purchasing, later handling the invoice and payment processes again. This point enhanced the business level communication with workflow information. There are many different workflow interfaces are available. SAP as our example is using the WF-XML.

Summarizing this section we are just facing another wave again where not the integration is in the focus, but the separation. If we consider from a bird perspective at the time of the first single process systems we had many separated applications, after the MRP and ERP systems integrated everything, and with these solutions the separation came into importance again. From processes point of view we can deal with cross-component workflows, where the whole process is executed not only in one system, but in more systems, and the data and workflow starting information is shared.

VI. Services and Service Orientation

The last sections projected already that with distributed environment higher communication is needed and sometimes the workflows need cross-system capabilities.

For communication the IT world enveloped the EAI (Enterprise Application Integration) solutions, which provide an interface central for the system offering their own interface technology by adapters (or connectors). This idea made possible to handle the interface difference between systems produced by numerous vendors with an easy way. On the other hand as a central solution it has the possibility to offer the data coming from any system to the others as a service. The EAI system can of course convert the data since it has to communicate with different technologies, but verification, mapping and real content conversion using data manipulation functions are also available. The processes available in a system can be offered for others to use. Each of the vendors developed its own communication layer, where the services available were offered for external use. The EAI solutions made it possible to collect the different services offered via different protocols and provide them to the others via their own or a common, standard protocol. This capability made it easier to create in a system new applications, where some of the processes were not executed in the system itself, but through the EAI solution in another system. The services could be thread after each other into cross component workflow. In this scenario the services are in focus, as the name SOA (Service Oriented Architecture) suggests.

Service orientation changed the roles again. The best solution if the person, who knows the business and the processes can plan and "develop" the workflows. But generally these people cannot program. This duality leads to task separation in the development. There should be so-called service developers, who create backend functionalities without UIs (User Interface), so only parameters are given. The developed services execute different process tasks and can be linked after each other. The product owners should define the granularity of the tasks as well, not only the main purpose and functional specification. To create corresponding surfaces user interface elements should be developed as well. Depending on the approach the UI developers work in the portal based environment or in the system offering flexible web enabled UI technology. The UI developer's task is to create UI elements (e.g. table, a push button), and not to create the end-user surface! UI elements should be available on different devices (browser, mobile device) as well. To reach the above-mentioned best case, when the business expert puts the pieces together, the Service Oriented (SOA) environment should offer browser based, easy (code-free) maintainable, intuitive, rapid, model based tool. The business expert has to use the UI elements to design the surface and search for services offered to fulfill the business requirements. The model-based tools generally use meta data to make the design and deployment easier. SOA changed the roles and timing of the development, because the real applications are made at customer, after development time. The SOA applications not remain unchanged, there are not customizable, but flexible for any changes at any time. With SOA another integration wave tower again above the IT see, but it brought a higher and logical level of integration.

VII. Process management, process control

In the previous step (and section) the business processes where cut into smaller pieces to produce services, which can be offered by a system. Services can be offered via Internet as well, and service directories are also available to use as central place for common services.

The main point in a service-oriented architecture is the service repository, which collects and offers all the available services within the company produced by different system and solutions. The end-users are connected not to the back-end (service producer) application directly, but to the process (or workflow) engine. To create cross-component applications, so-called composites the service repository, the process engine and the users are needed.

The composite services push the workflows a bit forward, because above the SOA layer a central Enterprise Service Bus (ESB) using BPEL (Business Process Execution Language) is involved. From application developer, business expert and end-user point of view the applications, systems are not mentioned, only services offered by the ESB (collected in service repository) and process flows can be built from them, where the end-user is in the focus. The human interfaces should be design and linked to roles, but the background steps should be only called from the ESB. As we discussed in the previous section the roles are changing, not the simple and stable, well-known processes are important anymore, but the flexibility, reusability and easy business level design. As a real symphonic orchestra a central component should organize and control the tasks to provide business process management and composition for the new applications. The orchestration here is a centralized control mechanism to control service interaction activities. The technical partnership is defined here and the abstract model of the composite to describe how to execute the services.

VIII. Conclusion

The first Workflows, SOA and BMP changed the role of the people around IT. The developers and their task were changed many times to be closer and further from the business. The people from business logic side tried to understand programming and computer behaviors, to create the best solutions. The end-user concentrated to single processes, later tried to think about process flows. The business processes should be automated and distributed between parties to separate roles. Each of these pushed the IT solutions to make possible better and more flexible process flows.

The business process surfed on the IT integration waves from single process application to integrated real-time monolith systems (MRP, ERP), and afterwards by adding new directions with specialized solutions (like SRM, CRM, BI, etc.) the wave came down again to the separation valley. The service level integration through the SOA brought the BPM surf up again to the integrated solution level. Generally the IT goes in the integration direction, but a new separation or higher level can come again.

IX. Acknowledgement

This work was support by Hungarian Government's Baross Competition, and Budafilter 94 Ltd.

REFERENCES

[1] Igor Barbaric, “Design Patterns in Object-Oriented ABAP” *Galileo Press Bonn-Boston*, *2010*

[2] Rich Heilman, Thomas Jung, “Next Generation ABAP Development” *Galileo Press Bonn-Boston*, *2007*

[3] Baude, F., “A component-based orchestration management framework for multidomain SOA,” *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium.* Doublin:, 2011, ISBN: 978-1-4244-9219-0, pp. 1156-1163).

[4] Rosenberg A., Rosing, Chase G., Omar R., and Taylor J., “Applying Real-World BPM in an SAP Environment” *Galileo Press Bonn-Boston*, ISBN: 978-1-59229-343-8 *2011*

[5] Ma, Z., “BPEL Fragments for Modularized Reuse in Modeling BPEL Processes” *Networking and Services, 2009. ICNS '09. Fifth International Conference on.* Valencia:,, ISBN: 978-1-4244-3688-0, 2009. April pp. 63).